*Help us provide free content to the world by **donating today!***

Application Specific Instruction Set Processor

From Wikipedia, the free encyclopedia
(Redirected from ASIP)

Application Specific Instruction-Set Processor or (**ASIP**) is a methodology used in System-on-a-Chip design. It represents a compromise between ASIC and general purpose CPU. In ASIP, the instruction set provided by the core can be configured to fit the specific application. This "configurability" of the core provides a tradeoff between flexibility and performance. Usually, the core is divided into two parts: *static* logic which defines a minimum ISA and *configurable* logic which can be used to design new instructions. The configurable logic can be programmed either in the "field" in a similar fashion to FPGA or during the chip synthesis.

References

- Cong, J. and Fan, Y. and Han, G. and Zhang, Z.. "*Application-specific instruction generation for configurable processor architectures*". ACM Press New York, NY, USA.
- Wirthlin, M.J. and Hutchings, B.L.. "*A Dynamic Instruction Set Computer*". IEEE Computer Society Press.
- M. Jain and M. Balakrishnan and A. Kumar. "*ASIP Design Methodologies : Survey and Issues*". IEEE.
- Zebo Peng. Application Specific Instruction Set Processor Architecture (http://citeseer.ist.psu.edu/326812.html).

Retrieved from "http://en.wikipedia.org/wiki/Application_Specific_Instruction_Set_Processor"

Categories: Computer architecture | Central processing unit | Computer hardware | Gate arrays | Integrated circuits

- This page was last modified 10:54, 14 August 2007.
- All text is available under the terms of the GNU Free Documentation License. (See **Copyrights** for details.) Wikipedia® is a registered trademark of the Wikimedia Foundation, Inc., a U.S. registered 501(c)(3) tax-deductible nonprofit charity.

Help us provide free content to the world by donating today!

Application-specific integrated circuit

From Wikipedia, the free encyclopedia

An **application-specific integrated circuit** (ASIC) is an integrated circuit (IC) customized for a particular use, rather than intended for general-purpose use. For example, a chip designed solely to run a cell phone is an ASIC. In contrast, the 7400 series and 4000 series integrated circuits are logic building blocks that can be wired together for use in many different applications. Intermediate between ASICs and standard products are *application specific standard products (ASSPs)*.

As feature sizes have shrunk and design tools improved over the years, the maximum complexity (and hence functionality) possible in an ASIC has grown from 5,000 gates to over 100 million. Modern ASICs often include entire 32-bit processors, memory blocks including ROM, RAM, EEPROM, Flash and other large building blocks. Such an ASIC is often termed a SoC (System-on-a-chip). Designers of digital ASICs use a hardware description language (HDL), such as Verilog or VHDL, to describe the functionality of ASICs.

Field-programmable gate arrays (FPGA) are the modern day equivalent of 7400 series logic and a breadboard, containing programmable logic blocks and programmable interconnects that allow the same FPGA to be used in many different applications. For smaller designs and/or lower production volumes, FPGA's may be more cost effective than an ASIC design. The Non-recurring engineering cost (the cost to set up the factory to produce a particular ASIC) can run into hundreds of thousands of dollars.

The general term application specific integrated circuit includes FPGAs, but most designers use ASIC only for non field programmable devices (e.g. standard cell or sea of gates) and make a distinction between ASIC and FPGAs.

Contents

History

The initial ASICs used gate array technology. Ferranti produced perhaps the first gate-array, the ULA (Uncommitted Logic Array), around 1980. Customization occurred by varying the metal interconnect mask. ULAs had complexities of up to a few thousand gates. Later versions became more generalized, with different base dies customised by both metal and polysilicon layers. Some base dies include RAM elements.

Standard cell design

In the mid 1980s a designer would choose an ASIC manufacturer and implement their design using the design tools available from the manufacturer. While third party design tools were available, there was not an effective link from the third party design tools to the layout and actual semiconductor process performance characteristics of the various ASIC manufacturers. Most designers ended up using factory specific tools to complete the implementation of their designs. A solution to this problem that also yielded a much higher density device was the implementation of Standard Cells. Every ASIC manufacturer could create functional blocks with known electrical characteristics, such as propagation delay, capacitance and inductance, that could also be represented in third party tools. Standard Cell design is the utilization of these functional blocks to achieve very high gate density and good electrical performance. Standard cell design fits between Gate Array and Full Custom design in terms of both its NRE (Non-Recurring Engineering) and recurring component cost.

By the late 1980s, logic synthesis tools became available. Such tools could compile HDL descriptions into a gate-level netlist. This enabled a style of design called standard-cell design. Standard-cell Integrated Circuits (ICs) are designed in the following conceptual stages, although these stages overlap significantly in practice.

These steps, implemented with a level of skill common in the industry, almost always produce a final device that correctly implements the original design, unless flaws are later introduced by the physical fabrication process.

1. A team of design engineers starts with a non-formal understanding of the required functions for a new ASIC, usually derived from Requirements analysis.
2. The design team constructs a description of an ASIC to achieve these goals using an HDL. This process is analogous to writing a computer program in a high-level language. This is usually called the RTL (Register transfer level) design.
3. Suitability for purpose is verified by functional verification. This may include such techniques as logic simulation, formal verification, emulation, or creating an equivalent pure software model (see Simics, for example). Each technique has advantages and disadvantages, and often several methods are used.
4. Logic synthesis transforms the RTL design into a large collection of lower-level constructs called standard cells. These constructs are taken from a standard-cell library consisting of pre-characterized collections of gates (such as 2 input nor, 2 input nand, inverters, etc.). The standard cells are typically specific to the planned manufacturer of the ASIC. The resulting collection of standard cells, plus the needed electrical connections between them, is called a gate-level netlist.
5. The gate-level netlist is next processed by a placement tool which places the standard cells onto a region representing the final ASIC. It attempts to find a placement of the standard cells, subject to a variety of specified constraints.
6. The routing tool takes the physical placement of the standard cells and uses the netlist to create the electrical connections between them. Since the search space is large, this process will produce a "sufficient" rather than "globally-optimal" solution. The output is a file which can be used to create a set of photomasks enabling a semiconductor fabrication facility (commonly called a 'fab') to produce physical ICs.

7. Given the final layout, circuit extraction computes the parasitic resistances and capacitances. In the case of a digital circuit, this will then be further mapped into delay information, from which the circuit performance can be estimated, usually by static timing analysis. This, and other final tests such as design rule checking and power analysis (collectively called signoff) are intended to insure that the device will function correctly over all extremes of the process, voltage and temperature. When this testing is complete the photomask information is released for chip fabrication.

These design steps (or flow) are also common to standard product design. The significant difference is that Standard Cell design uses the manufacturer's cell libraries that have been used in potentially hundreds of other design implementations and therefore are of much lower risk than full custom design. Standard Cells produce a design density that is cost effective, and they can also integrate IP cores and SRAM (Static Random Access Memory) effectively, unlike Gate Arrays.

Gate array design

Gate Array design is a manufacturing method in which the diffused layers, i.e. transistors and other active devices, are predefined and wafers containing such devices are held in stock prior to metallization, in other words, unconnected. The physical design process then defines the interconnections of the final device. For most ASIC manufacturers, this consists of from two to as many as five metal layers, each metal layer running perpendicular to the one below it. Non-recurring engineering costs are much lower as photo-lithographic masks are required only for the metal layers, and production cycles are much shorter as metallization is a comparatively quick process. **It is also important to the designer that minimal propagation delays can be achieved in ASICs versus the FPGA solutions available in the marketplace.** Gate Array ASICs are always a compromise as mapping a given design onto what a manufacturer held as a stock wafer never gives 100% utilization. Often difficulties in routing the interconnect require migration onto a larger array device with consequent increase in the piece part price. These difficulties are often a result of the layout software used to develop the interconnect.

Pure, logic-only Gate Array Design is rarely implemented by circuit designers today, replaced almost entirely by field programmable devices, such as FPGAs (Field Programmable Gate Arrays), which can be programmed by the user and thus offer minimal tooling charges (Non-recurring engineering (NRE)), marginally increased piece part cost and comparable performance. Today Gate Arrays are evolving into Structured ASICs that consist of a large IP core like a processor, DSP unit, peripherals, standard interfaces, integrated memories SRAM, and a block of reconfigurable uncommited logic. This shift is largely because ASIC devices are capable of integrating such large blocks of system functionality and "system on a chip" requires far more than just logic blocks.

The term "Gate Array" is almost synonymous and interchangeable with the term "Semi-Custom". Which term you would use depends on who you are; if you are a process engineer, more likely than not you would use "Semi-Custom" whereas if you are a logic (or gate level) designer, "Gate-Array" would probably be your term of choice.

Full-custom design

By contrast, Full-Custom ASIC Design defines all the photo lithographic layers of the device. Full

Custom Design is used for both ASIC design and for Standard Product design.

The benefits of Full-Custom Design usually include **reduced area** (and therefore recurring component cost), performance improvements and also the ability to integrate (include) analog components and other pre-designed (and thus fully verified) components such as microprocessor cores that form a System-On-Chip. **The disadvantages of Full-Custom can include increased manufacturing and design time, increased non-recurring engineering costs, more complexity in the Computer Aided Design (CAD) system and a much higher skill requirement on the part of the design team.** However for digital only designs, "standard-cell" cell libraries together with modern CAD systems can offer considerable performance/cost benefits with low risk. Automated layout tools are quick and easy to use and also offer the possibility to "hand-tweak" or manually optimise any performance limiting aspect of the design.

Structured/platform design

Structured (also referred to as Platform) ASIC Design is an ambiguous expression, with different meanings in different contexts. This is a relatively new term in the industry, which is why there is some variation in its definition. However, the basic premise of a Structured/Platform ASIC is that both manufacturing cycle time and design cycle time are reduced compared to cell-based ASIC by virtue of **there being pre-defined metal layers (thus reducing manufacturing time) and pre-characterization of what is on the silicon (thus reducing design cycle time).** One definition states that *In a "structured ASIC" design, the logic mask-layers of a device are predefined by the ASIC vendor (or in some cases by a third party). Design differentiation and customization is achieved by creating custom metal layers that create custom connections between predefined lower-layer logic elements. "Structured ASIC" technology is seen as bridging the gap between field-programmable gate arrays and "standard-cell" ASIC designs. Because only a small number of chip layers must be custom-produced, "structured ASIC" designs have much smaller non-recurring expenditures (NRE) than "standard-cell" or "full-custom" chips, which require that a full mask set be produced for every design.* This is effectively the same definition as a Gate Array.

What makes a Structured/Platform ASIC different from a gate array is that in a gate array the predefined metal layers serve to make manufacturing turnaround faster. In a Structured/Platform ASIC the predefined metallization is primarily to reduce cost of the mask sets and is also used to make the design cycle time significantly shorter as well. For example, in a cell-based or gate-array design the user often must design power, clock, and test structures themselves; these are predefined in most Structured/Platform ASICs and therefore can save time and expense for the designer compared to gate-array. Likewise, the design tools used for Structured/Platform ASIC can be substantially lower cost and easier (faster) to use than cell-based tools, because the tools do not have to perform all the functions that cell-based tools do. In some cases, the Structured/Platform ASIC vendor requires that customized tools for their device (for example, custom physical synthesis) be used, also allowing for the design to be brought into manufacturing more quickly. ChipX, Inc. and eAsic are examples of vendors offering this kind of Structured ASIC.

One other important aspect about Structured/Platform ASIC is that it allows IP that is common to certain applications or industry segments to be "built in", rather than "designed in". By building the IP directly into the architecture the designer can again save both time and money compared to designing IP into a cell-based ASIC.

The best advice is to read carefully how the vendor defines its particular Structured or Platform ASIC, as there are significant differences between vendors offering these devices.

The Altera technique of producing a structured cell ASIC where the cells are the same design as the FPGA, but the programmable routing is replaced with fixed wire interconnect is called HardCopy.[1] These devices then do not need re-programming and cannot be re-programmed as an FPGA.

The Xilinx technique of producing a customer specific FPGA, that is 30% - 70% less expensive than a standard FPGA and where the cells are the same as the FPGA but the programmable capability is removed, is called EasyPath.[1]

Cell libraries, IP-based design, hard and soft macros

Cell Libraries of logical primitives are usually provided by the device manufacturer as part of the service. Although they will incur no additional cost, their release will be covered by the terms of a non disclosure agreement (NDA) and they will be regarded as intellectual property by the manufacturer. Usually their physical design will be pre-defined as so they could be termed hard macros.

But what most engineers understand as "intellectual property" are IP cores, designs purchased from a third party as sub-components of a larger ASIC. They may be provided as an HDL description (often termed a soft macro), or as a fully routed design that could be printed directly onto an ASIC's mask (often termed a hard macro). Many organizations now sell such pre-designed IP, and larger organizations may have an entire department or division to produce such IP for the rest of the organization. For example, one can purchase CPUs, ethernet, USB or telephone interfaces. Indeed, the wide range of functions now available is a significant factor in the phenomenal increase in electronics in the late 1990s and early 2000s; as intellectual property takes a lot of time and investment to create, its re-use and further development cuts product cycle times dramatically and creates better products.

Soft Macros are often process independent; i.e., they can be fabricated on a wide range of manufacturing processes and indeed different manufacturers.

Hard Macros are process limited and usually further design effort must be invested to migrate (port) to a different process or manufacturer.

Multi-project wafers

Some manufacturers offer Multi-Project Wafers (MPW) as a method of obtaing low cost prototypes. Often called shuttles, these MPW, containing several designs, run at regular, scheduled intervals on a "cut and go" basis, usually with very little liability on the part of the manufacturer. The contract involves the assembly and packaging of a handful of devices. The service usually involves the supply of a physical design data base i.e. masking information or Pattern Generation (PG) tape. The manufacturer is often referred to as a "silicon foundry" due to the low involvement it has in the process. See also Multi Project Chip.

ASIC designers

- Spike Technolgies (Bangalore, India; California, US)

ASIC manufacturers (foundries)

- Chartered
- cPackets
- Fujitsu
- Freescale
- IBM
- Infineon Technologies
- MHS Electronics
- MOSIS
- NVIDIA
- NEC
- Samsung
- SMIC
- Texas Instruments
- TSMC
- UMC
- X-Fab

References

1. ^ *a* *b* Richard Ball. "The promise of structured Asic (http://www.electronicsweekly.com/Articles/2004/10/26/33416/The+promise+of+structured+Asic. *Electronics Weekly*, 2004-10-26.

Further reading

- Paul Naish (1988). Designing ASICs (http://web.ukonline.co.uk/paul.naish/DA/contents.htm). — An Introduction to ASIC design with an emphasis on synchronous clocking techniques. Written within the context of a training department. Perhaps rather dated now, as it deals only with primitive logic. Analogue engineers who need to include some digital logic into their designs would find this particularly useful.
- Kevin Morris. "Cost-Reduction Quagmire: Structured ASIC and Other Options (http://www.fpgajournal.com/articles/20041123_quagmire.htm)", *FPGA and Programmable Logic Journal*, 2003-11-23.
- Jim Turley. "Hard Choices Among FPGA Hardening Options (http://www.techonline.com/article/192200241)", *TechOnline*, 2005-04-07.
- Anthony Cataldo. "Xilinx looks to ease path to custom FPGAs (http://www.eetimes.com/story/OEG20020325S0060)", *EE Times*, CMP Media, LLC, 2002-03-26.
- "Xilinx intros next-gen EasyPath FPGAs priced below structured ASICs (http://findarticles.com/p/articles/mi_m0GZQ/is_41_45/ai_n8968679)", *EDP Weekly's IT Monitor*, Millin Publishing, Inc., 2004-10-18.
- Golshan, K. (2007). *Physical design essentials: an ASIC design implementation perspective.* New York: Springer. ISBN 0387366423.

See also

- FPGA: Field-programmable gate array
- Electronic design automation
- System-on-a-chip

External links

- XML on a chip? (http://www.ximpleware.com/wp_SUN.pdf)
- HardCopy (http://www.altera.com/products/devices/hardcopy/hrd-index.html) Altera's structured ASIC
- EasyPath (http://www.xilinx.com/EasyPath) Xilinx' EasyPath Solution and Alternative to Structured ASICs for moving FPGA technology to very High Volume production
- LinearChip (http://www.linearchip.com/) Linearchip partners with foundries to create analog, mixed signal and custom ASICs for commercial and military applications
- Hybrid ASIC (http://www.chipx.com/) ChipX has Structured ASIC, Embedded Array, and standard Cell ASIC with seamless conversion between types.

Retrieved from "http://en.wikipedia.org/wiki/Application-specific_integrated_circuit"

Categories: All articles with unsourced statements | Articles with unsourced statements since February 2007 | Gate arrays | Integrated circuits

- This page was last modified 16:54, 25 September 2007.
- All text is available under the terms of the GNU Free Documentation License. (See **Copyrights** for details.) Wikipedia® is a registered trademark of the Wikimedia Foundation, Inc., a U.S. registered 501(c)(3) tax-deductible nonprofit charity.